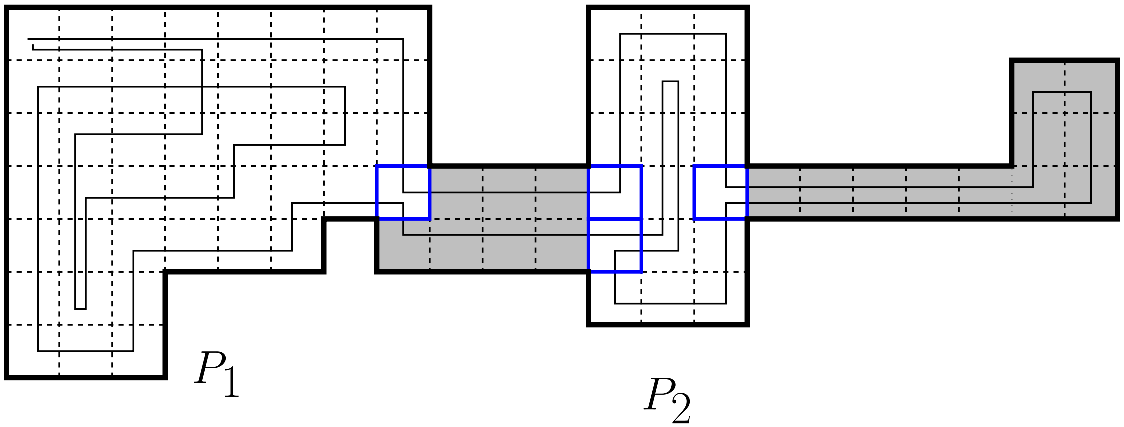
Online Motion Planning MA-INF 1314
Smart DFS

Elmar Langetepe
University of Bonn

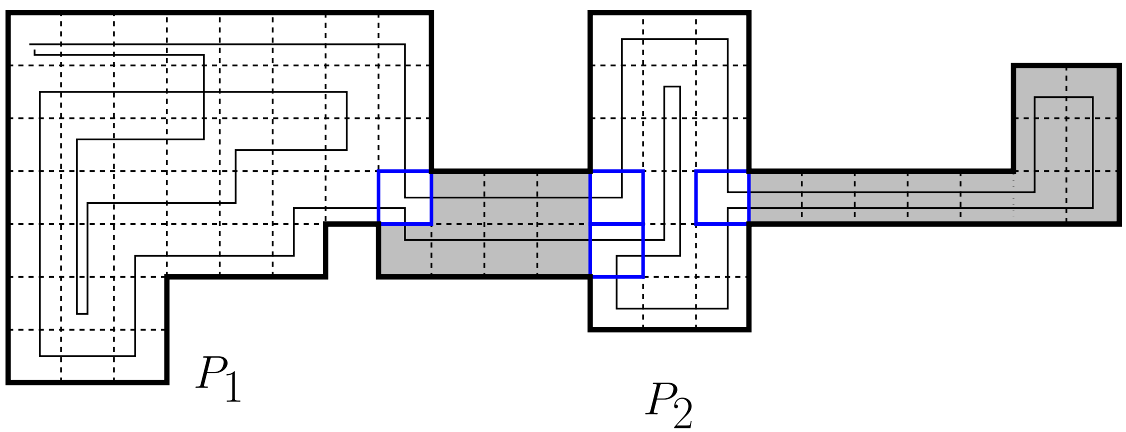
Repetition!

- SmartDFS: DFS Return path and component opt.
- SmartDFS: Shortest return path
- Wavefron Algorithmu (Lee): $O(n)$, n cells
- Comp. Factor: $S(P) \leq \frac{4}{3} C(P) - 2$ (Untere Schranke $\frac{7}{6}$)
- Observation: Optimally in narrow passages!



Wiederholung!

- Analyse polygons P_i, $i = 1, \ldots, k$
- Induction over split-cells
- Induction-Basesanfang: No split-cell in layer 1.
- **Lemma** $E(P) \leq \frac{2}{3}C(P) + 6$ Backward analysis
- **Lemma** $S(P) \leq C(P) + \frac{1}{2}E(P) - 5$, Two steps less by Offsetlemma!
- Kombination gives Iduction-Base!

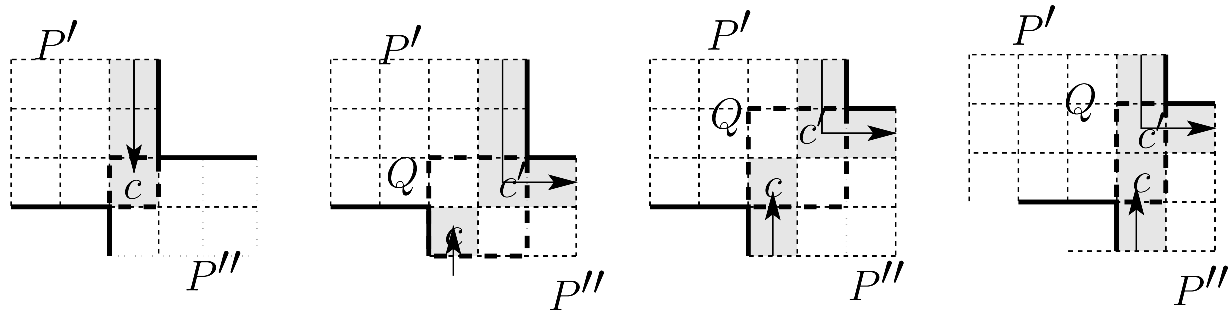


Induction-Step: Fall (i): $S(P_i) \leq \frac{4}{3}C(P_i) - 2$

- $S(P_i) = S(P') + S(P'')$ (Gate) $C(P_i) = C(P') + C(P'') - 1$
- IH. for P' and P''

$$S(P_i) = S(P') + S(P'') \leq \frac{4}{3}C(P') - 2 + \frac{4}{3}C(P'') - 2$$

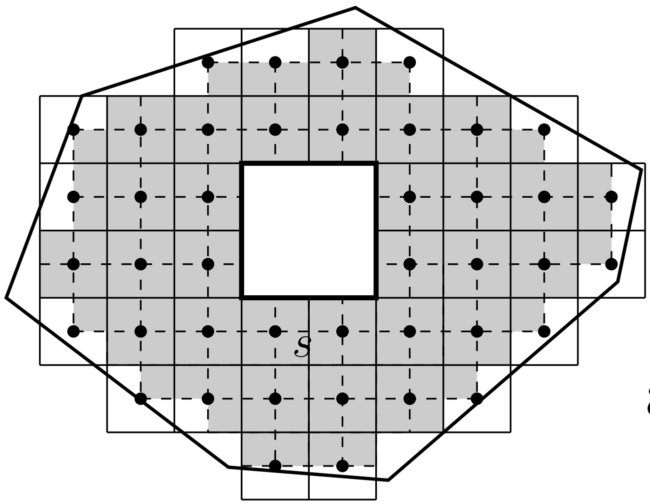
$$\leq \frac{4}{3}C(P_i) + \frac{4}{3} - 4 < \frac{4}{3}C(P_i) - 2$$

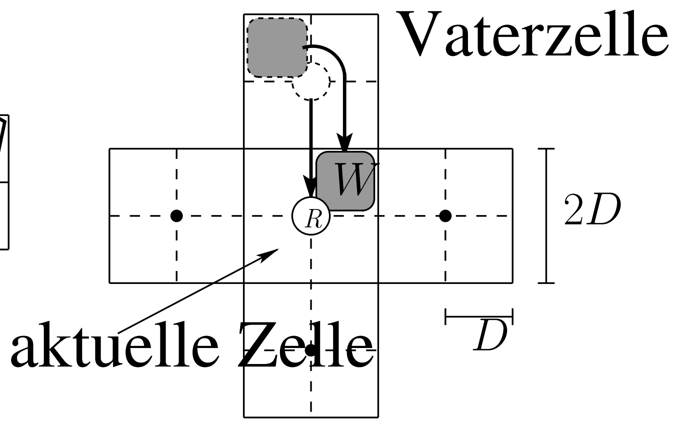


(i) (ii) (iii) (iv)

General gridpolygons

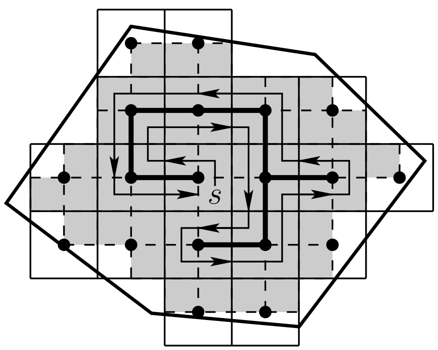
- Change the modell, due to the analysis
- $2D$ cells with center, sub-cells
- *See* adjacent $2D$ Zellen
- Tool W of size D



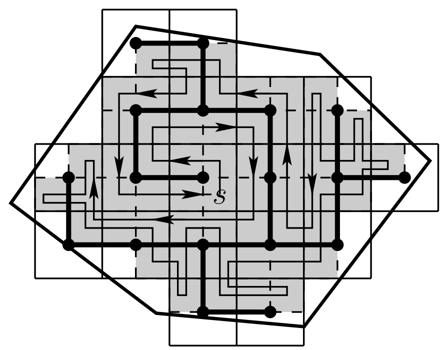


Spanning Trees

- Online mit DFS with a Spanning Tree of 2D vertices
- Move the tool along the tree
- Left-Hand Rule along the tree
- Variants 2D cell totally free for the edge/or not!
- Any cell only once or more than once



Nur unbelegte 2D Zellen



Belegt aber begehbare 2D Zellen

2D Spiral STC: 2DSPSTC($parent, current$)

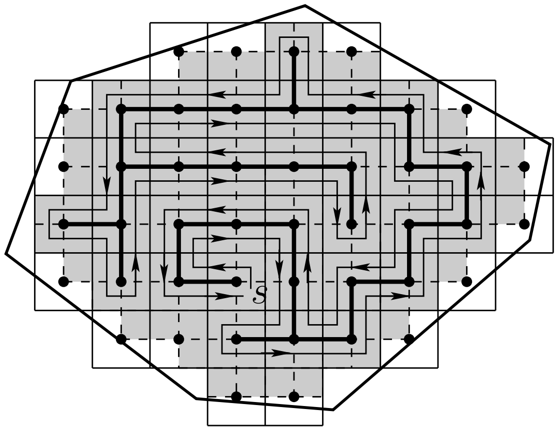
Mark *current* as explored
while *current* has unexplored neighbour **do**

- Search from *parent* in ccw order
  neighbour *free* non-explored/free
- Span.Tree edge *current* zu *free*.
- Move tool L-H-R along
  Span.Tree edge to
  first sub-cell of *free*
- 2DSPSTC(*current*, *free*)

end while
if *current* $\neq s$ **then**

- From *current* by L-H-R along
  Span.Tree to subcell of *parent*

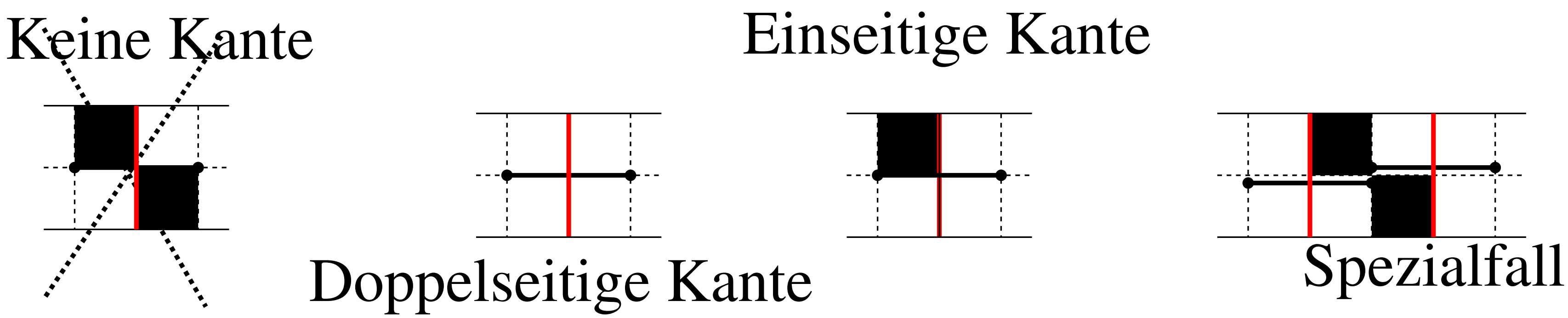
end if



Nur freie 2D Zellen

Partially blocked 2D cells

- Spanning Tree, edge is free/not the full cell
- Reachable D sub-cells ?
- Different types
- **Definition:** double-sided edge, one-sided edge



2D Spiral STC: $\mathbf{2DSPSTC}(\boldsymbol{parent}, \boldsymbol{current})$

Mark *current* as explored

while *current* has unexplored neighbour **do**

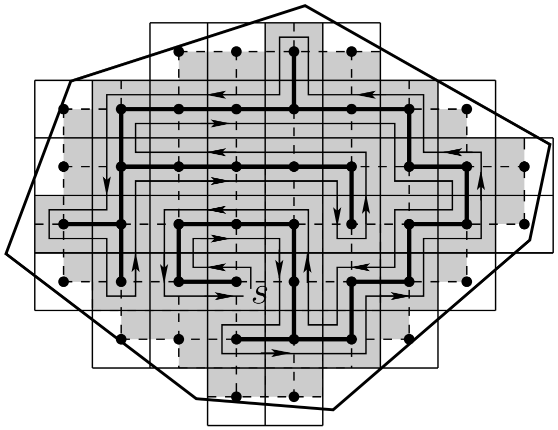
- Search from *parent* in ccw order
  neighbour *free* non-explored/free
- Span.Tree edge *current* zu *free*.
- Move tool L-H-R along
  Span.Tree edge to
  first sub-cell of *free*
- 2DSPSTC(*current*, *free*)

end while

if *current* $\neq s$ **then**

- From *current* by L-H-R along
  Span.Tree to subcell of *parent*

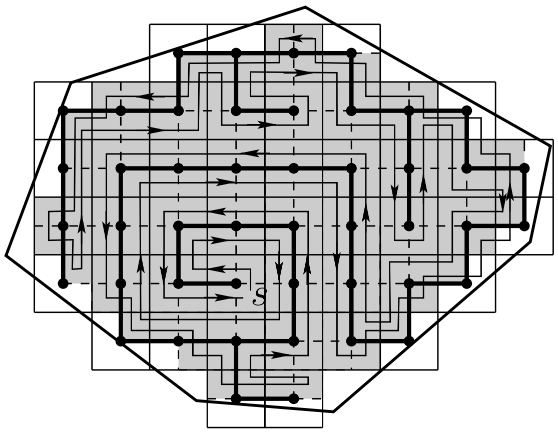
end if



Nur freie 2D Zellen

Spiral STC: $\mathrm{SPSTC}(\boldsymbol{parent}, \boldsymbol{current})$

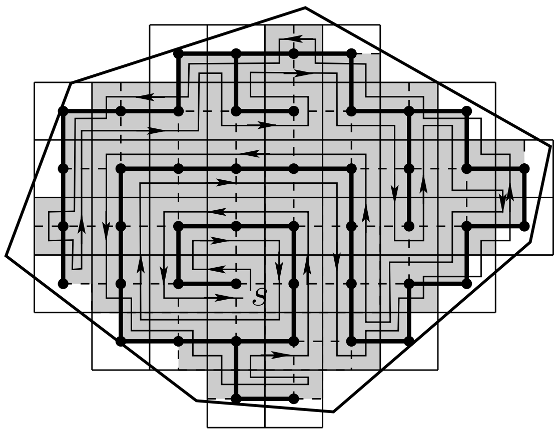
- Search from *parent* in ccw order neighbour *free* non-explored, s.th. Spanning edge can be build (migth be single-sided)
- Search from *parent* in ccw order neighbour *free* non-explored/free



Falls Knoten erreichbar

Spiral STC: $\mathrm{SPSTC}(\boldsymbol{parent}, \boldsymbol{current})$

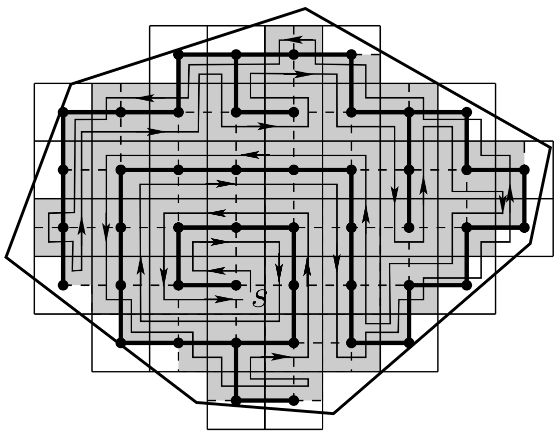
- Move tool along the spanning tree edge to the first reachable sub-cell of *free*. Left-Hand-Rule for double-sided edges. Avoid obstacles of single-sided edges. Tool might change to the left of the spanning tree edge.
- Move tool L-H-R along Span.Tree edge to first sub-cell of *free*



Falls Knoten erreichbar

Spiral STC: $\mathbf{SPSTC}(\boldsymbol{parent}, \boldsymbol{current})$

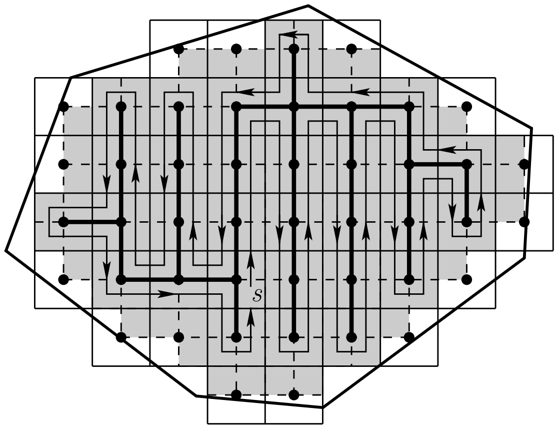
- Move tool along the spanning tree edge to the first reachable sub-cell of *free*. Left-Hand-Rule for double-sided edges.
  Avoid obstacles of single-sided edges.
  Tool might change to the left of the spanning tree edge.
- From *current* by L-H-R along Span.Tree to subcell of *parent*

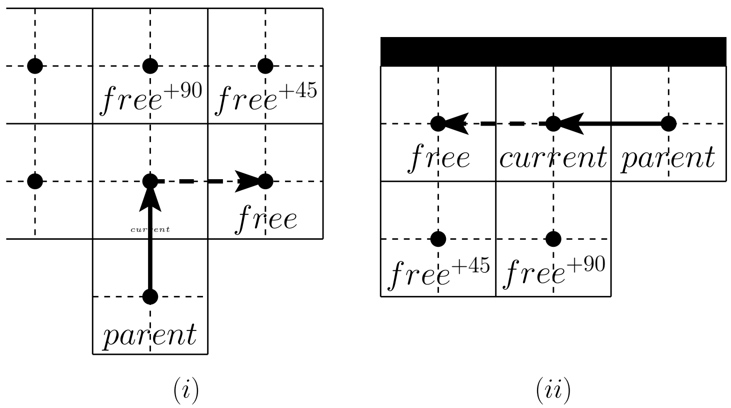


Falls Knoten erreichbar

Less rotations for the tool

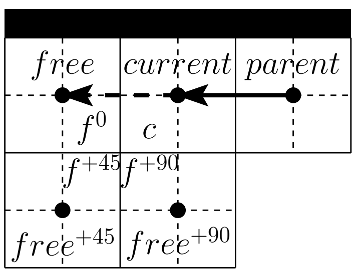
- Avoid spiral-like paths
- Move in columns
- Scan also diagonally adjacent 2D cells
- ScanSTC 2D Algorithm
- Also for the Backtracking step

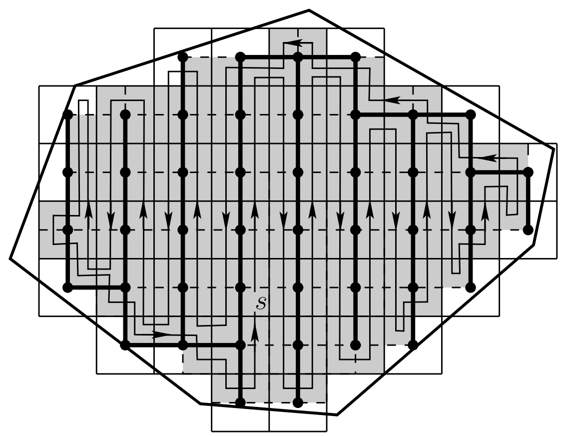




Less rotations for the tool

- Avoid spiral-like paths
- Move in columns
- Also for the general case/path should exist
- Scan also diagonally adjacent 2D cells
- ScanSTC Algorithm
- Also for the Backtracking step





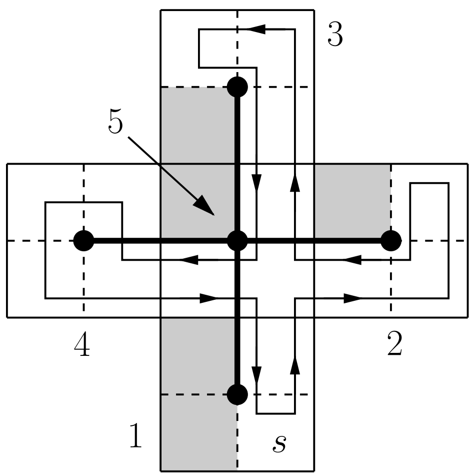
Analysis! Theorem

- General Spiral STC
- Number of steps for the tool
- As given by SmartDFS, C plus overhead
- D sub-cells, at the *boundary* K in total

P gridpolygon, C reachable sub-cells. K reachable sub-cells that are diagonally adjacent to a blocked sub-cell. P is explored by Spiral-STC or Scan-STC. Requires $O(C)$ space and $O(C)$ time. The number of steps for the tool is restricted to $S \leq C + K$.

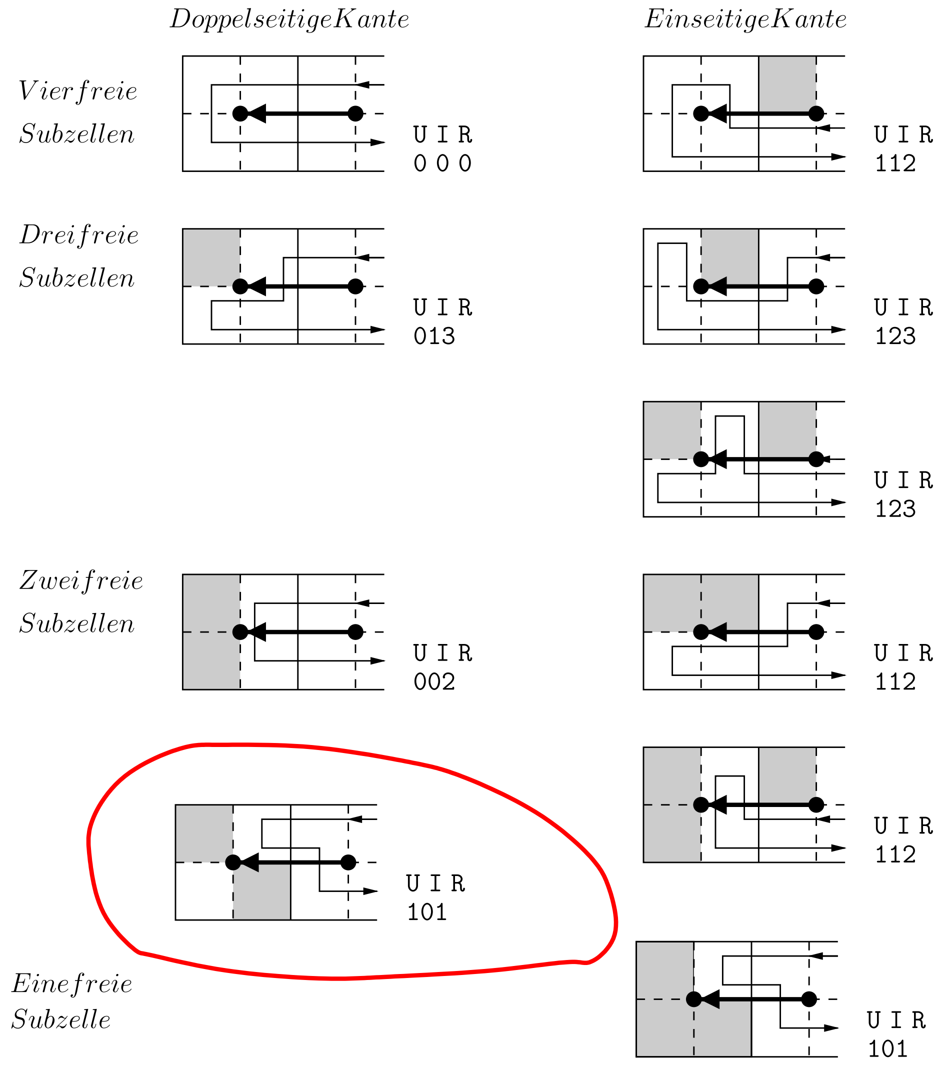
Number of steps! Example!

- Locally, count boundary sub-cells
- Local analysis, multiple visits, charge boundary sub-cells
- 2D Inner-cell/ Intra-cell
- Systematically: Boundary sub-cells charged by *Inner* plus *Intra*



Zelle	Übergr.	Intern	Gesamt	Randzellen
1	0	1	1	2
2	1	2	3	3
3	1	2	3	3
4	1	1	2	2
5	1	2	3	3

Number of steps! Theorem Systematically!

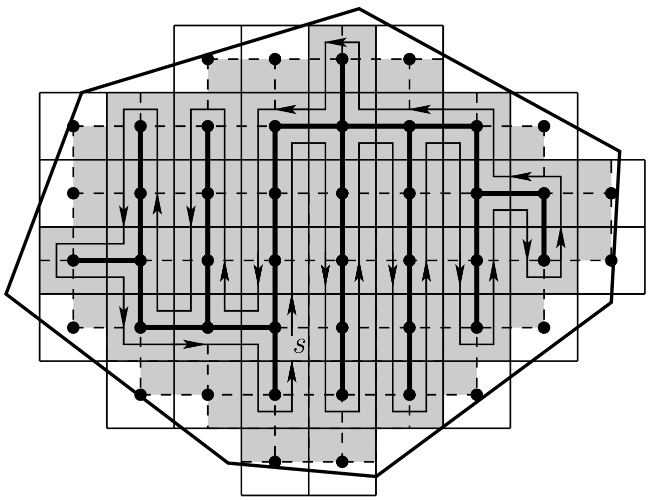


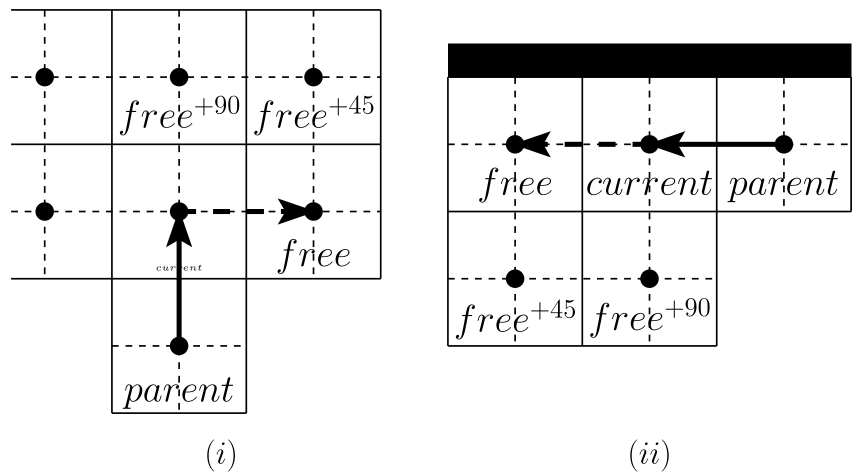
Running time and space required Theorem

- $C + K$ steps but K is bounded by C
- Local decisions: $O(1)$
- Running time and space $O(C)$

Analysis of 2D-ScanSTC

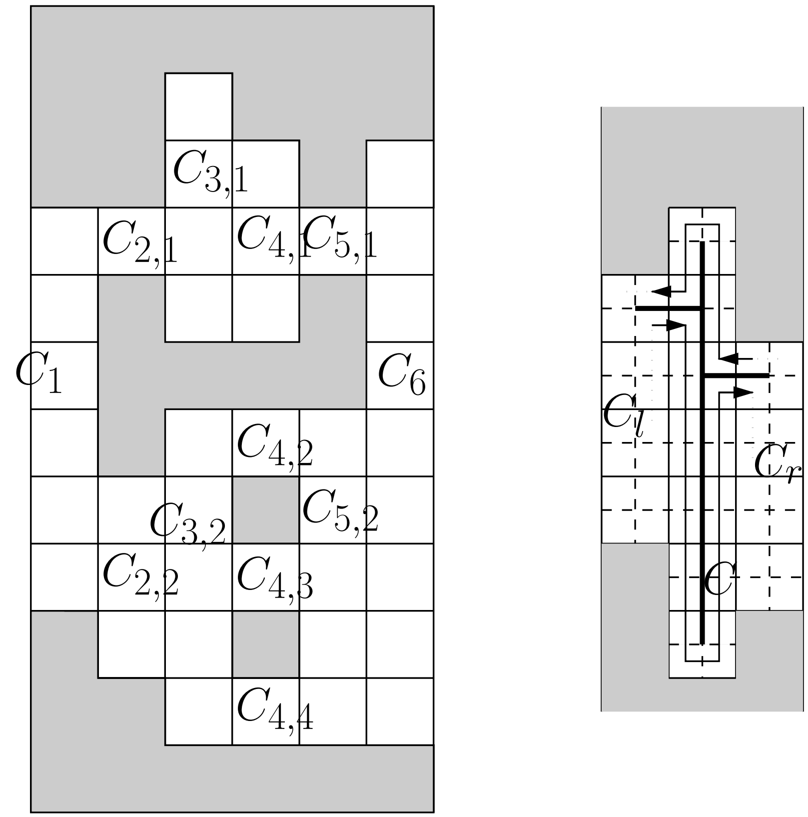
- Give a Scan-Preference: I.e. Vertically
- Decide only locally (more information)
- How many *bad* horizontal edges?
- Optimal numberl H_{opt}
- Compare with 2D-ScanSTC: Say H_{STC}





Analysis of 2D-ScanSTC

- Columns connectivity
- From Left to Right X nach Y
- Sum up the Differences: Overall Z
- Connectivity changes



(i) (ii)

Proof Sketch

- H_{Opt} optimal number of horizontal edges in the spanning tree. Z number of connectivity changes of P. $2D$-Scan-STC requires

$$H_{STC} \leq H_{Opt} + Z + 1$$

horizontal edges in its spanning tree.